A PREDICTIVE CODING MODEL OF HIPPOCAMPO-NEOCORTICAL INTERACTIONS INVOLVED IN MEMORY REPLAY

Anonymous authors
Paper under double-blind review

ABSTRACT

The neocortex and the hippocampus are two complementary learning systems which interact during memory construction and consolidation. The hippocampus stores episodic memories coming from the neocortex passing through the entorhinal cortex, and later replays them back to the neocortex to transform them into semantic memory during memory consolidation. It is thought that memory replay is a generative process, involved in imagining, because new episodes can also be generated and instantiated in the neocortex. Here we present a computational model of hippocampal-neocortical interactions based on a predictive coding network with two hidden layers, which are mapped onto the visual cortex and the entorhinal cortex. Improving on a previous implementation of this network, our simulations provide a mechanistic account of memory replay in the neocortex.

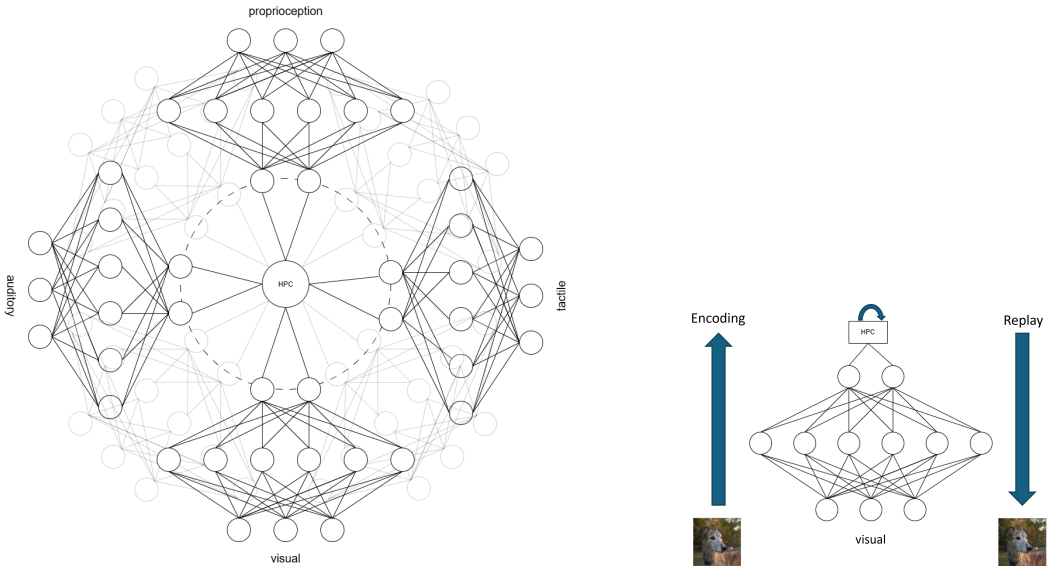
1 INTRODUCTION

According to the complementary learning systems theory, the necortex is responsible for semantic memory, that is the general knowledge that we have about the world, whereas hippocampus stores episodic memories, which correspond to an individual's emotional and sensory experiences (Kumaran et al., 2016). For example, the experience of encountering a particularly odd-looking dog (his look, bark, smell and the surprise you felt when seeing it) can be stored as an episodic memory whereas the knowledge about what characterizes a typical dog is semantic memory.

The episodic memories stored in the hippocampus are replayed during rest or sleep, reactivating the corresponding activity in the neocortex so that they are gradually integrated in the semantic memory of the neocortex. This idea is supported by empirical evidence from rodent studies during spatial navigation, where it was found that the rodent hippocampus generates sequences of activations during wakeful rest or sleep that reflect past trajectories (Buzsáki, 2015). In machine learning, experience replay has been shown to prevent catastrophic forgetting in a continual learning setting, where the learning of new tasks interferes with the knowledge of previously learned tasks. It consists of continually storing episodes in a memory buffer and replaying them when learning a new task.

It was later found that the internally generated hippocampal sequences are not merely replays of past trajectories, but also include paths that were never experienced before (Kumaran et al., 2016). This has prompted researchers from computational neuroscience and brain-inspired machine learning to hypothesize that the hippocampus is a generative model and that memory replay is a generative process, refered to as generative replay (Stoianov et al., 2022; van de Ven et al., 2020). Moreover, generative replay has been shown to improve the performance of reinforcement learning agents over experience replay Wang et al. (2025).

Because of the similarity of architecture in all its areas, researchers hypothesized that a common algorithm underlies computations in the neocortex (Friston, 2003; Hawkins et al., 2019). Principles of organization in the neocortex have emerged from empirical studies in the visual cortex, which have shown that this region is arranged hierarchically, with forward connections from lower to higher areas, and backward connections from higher to lower areas (Friston, 2003). At the apex of the neocortical hierarchy, the hippocampus receives input from the entorhinal cortex, which combines representations from different high-level neocortical areas of different sensory modalities (Barron et al., 2020), as illustrated in Figure 1a. Therefore, replay could drive neocortical activity using



(a) Neocortical hierachy (b) Replay in visual cortex

Figure 1: (a) The high-level representations in neocortical areas corresponding to different sensory modalities are combined in the entorhinal cortex (represented as the dotted circle) which is the input of the hippocampus (HPC), located at the apex of the neocortical hierarchy. (b) Experience replay is illustrated by isolating the visual cortex from (b). An image is processed by the visual cortex before being stored in the hippocampus (HPC). It can later be replayed by the hippocampus in the visual cortex.

backward connections from the entorhinal cortex to the neocortical hierachy, as illustrated for the visual cortex in Figure 1b.

In contrast, modern artificial neural networks used in image recognition only have forward connections. Recent work showed that adding feedback connections to recurrent vision models provides robustness to noise perturbations and adversarial attacks when coupled with stochastic neural variability during training Greco et al. (2025). However, this kind of models is specific to image classification, and does not intend to capture generative processes such as visual imagery. Generative models such as Variational Autoencoders (VAE) capture this kind of generative processes using decoders which can be seen as backward connections, but the neurons in the network are updated in a feedforward fashion. In contrast, activity of neurons in the neocortex is the result of the interaction of feedforward and feedback information. In addition, learning in these artificial neural networks is based on backpropagation, which is thought to be biologically implausible.

Rooted in studies of the visual cortex, predictive coding has been proposed in computational neuroscience as a general theory of cortical computation, which maps well to the neocortex in terms of architecture and information processing, as well as local learning rule (Friston, 2003). This framework has been implemented in different versions to solve practical tasks in machine learning, both supervised and unsupervised (Pinchetti et al., 2025; Ororbia & Kifer, 2022). However, little work has been done in computational neuroscience to study how this theory can account for the functions of the neocortex, in a biologically plausible way. Recently, (Fontaine & Alexandre, 2025) proposed a biologically inspired model of the neocortex, based on a predictive coding network with two hidden layers and showed that it could account for perceptual and mnesic mechanisms, including experience replay and recall. However, the study was focused on memory retrieval from a corrupted input because the aim was to show that the neocortex could not recall the details of episodic memories like the hippocampus does. Thus, the model was trained on only two classes of digits from the MNIST dataset, and the representations at the top of the hierarchy were overlapping at the boundary of the two classes, resulting in imperfect experience replays. In this paper, we tackle these limitations by accurately tuning the number of hidden units and stabilizing the convergence of the model on the full MNIST dataset using a learning rate scheduler. We found that adding more neurons in the second

hidden layer does not improve reconstruction (quite the contrary), but allows the episodic replays to be more accurate, by increasing the linear separability of representations corresponding to different classes at the top of the hierarchy. Furthermore, we show that the model can account for generative replay in addition to episodic replay.

2 Related works

In a bio-inspired model of memory consolidation (Spens & Burgess, 2024), the neocortex is modelled as a VAE, whereas the hippocampus corresponds to a memory buffer replaying episodic memories to the input of the VAE. Even though this model captures the cognitive process of memory consolidation, it is not faithful to the architecture and information processing of the corresponding structures in the brain. Other works studied the generation of inputs in predictive coding networks (PCN). On the one hand, Oliviers et al. (2024) proposed Monte Carlo predictive coding for learning probability distributions of sensory inputs, arguing that classical predictive coding demonstrated limited performance in generative tasks. On the other hand, preliminary work by Millidge (2019) showed that a PCN with one hidden layer can be used to generate inputs by sampling points close to the training data in the latent space, even though the generated samples were blurry and the authors did not describe their method for sampling. Ororbia & Kifer (2022) proposed to generate images from an extended version of PCN with ancestral sampling, but only three images per class are shown. In addition, using the model proposed by Friston (2003), the authors only show nearest neighbor samples that match an original data point for each class, leaving aside a large part of the image space which is covered by the generative model.

3 Predictive coding

Predictive coding networks (PCN) are based on hierarchical generative models with L layers, of the type

$$p_{\boldsymbol{\theta_0}}(\boldsymbol{h_0}) = \mathcal{N}(\boldsymbol{h_0}; \boldsymbol{\mu_0}, \boldsymbol{\Sigma_0}), \boldsymbol{\mu_0} = \boldsymbol{\theta_0}$$
$$\forall l \in \{1, 2, ..., L\}, p_{\boldsymbol{\theta_l}}(\boldsymbol{h_l} \mid \boldsymbol{h_{l-1}}) = \mathcal{N}(\boldsymbol{h_l}; \boldsymbol{\mu_l}, \boldsymbol{\Sigma_l}), \boldsymbol{\mu_l} = f_l(\boldsymbol{\mu_{l-1}}; \boldsymbol{\theta_l})$$

where $\boldsymbol{h} = (\boldsymbol{h_0}, \boldsymbol{h_1}, ..., \boldsymbol{h_L})$ are the states and $\boldsymbol{\theta} = (\boldsymbol{\theta_0}, \boldsymbol{\theta_1}, ..., \boldsymbol{\theta_L})$ are the parameters. Level L is the input level, so $\boldsymbol{h_L}$ is the input state and $\boldsymbol{h_0}, \boldsymbol{h_1}, ..., \boldsymbol{h_{L-1}}$ are the latent states. In practice, we use $\boldsymbol{\mu_0} = \mathbf{0}$ and $\boldsymbol{\Sigma_l} = \boldsymbol{I}$ for all $l \in \{0, 1, ..., L\}$. Therefore, the prior on the latent state of level 0 is a centered isotropic multivariate Gaussian $p_{\boldsymbol{\theta_0}}(\boldsymbol{h_0}) = \mathcal{N}(\boldsymbol{h_0}; \mathbf{0}, \boldsymbol{I})$ with no parameter.

Recognition is assumed to be deterministic, such that for an input $\boldsymbol{x}$,

$$q_{\boldsymbol{\phi}}(\boldsymbol{h_0}, \boldsymbol{h_1}, ..., \boldsymbol{h_{L-1}} \mid \boldsymbol{x}) = \prod_{l=0}^{L-1} \delta(\boldsymbol{h_l} - \boldsymbol{\phi_l})$$

where $\boldsymbol{\phi} = (\boldsymbol{\phi_0}, \boldsymbol{\phi_1}, ..., \boldsymbol{\phi_{L-1}})$ is an estimate of the latent states $\boldsymbol{h_0}, \boldsymbol{h_1}, ..., \boldsymbol{h_{L-1}}$ corresponding to the input $\boldsymbol{x}$.

Inference of the latent states for an input $\boldsymbol{x}$ results from the minimization of a lower bound to the negative likelihood, called variational free energy

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) &= -\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{h_0}, \boldsymbol{h_1}, ..., \boldsymbol{h_{L-1}} \mid \boldsymbol{x})} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{h_0}, \boldsymbol{h_1}, ..., \boldsymbol{h_{L-1}}, \boldsymbol{x})\right] \\
&= -\log p_{\boldsymbol{\theta}}(\boldsymbol{\phi_0}, \boldsymbol{\phi_1}, ..., \boldsymbol{\phi_{L-1}}, \boldsymbol{x}) \\
&= -\log p_{\boldsymbol{\theta_0}}(\boldsymbol{\phi_0}) - \log p_{\boldsymbol{\theta_1}}(\boldsymbol{\phi_1} \mid \boldsymbol{\phi_0}) - \log p_{\boldsymbol{\theta_2}}(\boldsymbol{\phi_2} \mid \boldsymbol{\phi_1}) - ... - \log p_{\boldsymbol{\theta_L}}(\boldsymbol{\phi_L} \mid \boldsymbol{\phi_{L-1}}) \\
&= -\log p(\boldsymbol{\phi_0}) + \frac{1}{2} \sum_{l=1}^{L} \left[\boldsymbol{\xi_l}^T \boldsymbol{\xi_l} + \log |\boldsymbol{\Sigma_l}|\right] + \text{constant}
\end{aligned}$$

where taking the logarithm of a Gaussian distribution results in a quantity

$$\boldsymbol{\xi_l} = \boldsymbol{\Sigma_l}^{-\frac{1}{2}} (\boldsymbol{\phi_l} - f_l(\boldsymbol{\phi_{l-1}}; \boldsymbol{\theta_l})). \tag{1}$$

which can be seen as a prediction error for layer l. Thus, the variational free energy corresponds to the sum of prediction errors in all layers, and PCNs learn hierarchical predictive representations of the input.

When presented with an input $\boldsymbol{x}$, the latent states are updated according to

$$\dot{\boldsymbol{\phi}}_l = -\nabla_{\boldsymbol{\phi}_l}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = -\frac{\partial \boldsymbol{\xi}_{l+1}^T}{\partial \boldsymbol{\phi}_l}\boldsymbol{\xi}_{\boldsymbol{l+1}} - \frac{\partial \boldsymbol{\xi}_l^T}{\partial \boldsymbol{\phi}_l}\boldsymbol{\xi}_l \tag{2}$$

until the variational free energy is minimized. In practice, we only update the latent states T times during training.

Similarly, learning of the parameters $\boldsymbol{\theta} = (\boldsymbol{\theta_1}, \boldsymbol{\theta_2}, ..., \boldsymbol{\theta_L})$ corresponds to the minimization of the variational free energy $F(\boldsymbol{\theta}) = \mathbb{E}_{p(\boldsymbol{x})}\left[\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}))\right]$. After several image presentations, the parameters are updated once following

$$\forall l \in \{1, 2, ..., L\}, \dot{\boldsymbol{\theta}}_l = -\nabla_{\boldsymbol{\theta}_l} F = -\mathbb{E}_{p(\boldsymbol{x})}\left[\frac{\partial \boldsymbol{\xi}_l^T}{\partial \boldsymbol{\theta}_l}\boldsymbol{\xi}_l\right]. \tag{3}$$

This algorithm can be implemented in a neural network, hence the name PCN, with only local computations for inference and learning. In a PCN, each level l consists of two types of neurons, with activity $\boldsymbol{\phi}_l$ and $\boldsymbol{\xi}_l$ respectively. When mapped to the neocortical hierarchy, level $l+1$ is the level below l, with level 0 at the top and level L at the bottom. From equation 1, it can be seen that neurons $\boldsymbol{\xi}_l$ compute the prediction errors, based on lateral connections with neurons $\boldsymbol{\phi}_l$ at the same level and inhibitory feedback connections with neurons $\boldsymbol{\phi}_{l-1}$ in the level above, which provide the predictions. Equation 2 shows that neurons $\boldsymbol{\phi}_l$ receive connections from error neurons in the same level $\boldsymbol{\xi}_l$ and the level below $\boldsymbol{\xi_{l+1}}$. In addition, it can be seen that equation 3 corresponds to Hebbian learning, as shown in the next section.

While we choose fixed covariances $\boldsymbol{\Sigma_l} = \boldsymbol{I}$, it was proposed in later publications that the inverse covariance, called precision, $\boldsymbol{\Sigma_l}^{-1}$ is predicted by higher layers and mediates attention (Feldman & Friston, 2010). Indeed, if precision $\boldsymbol{\Sigma_l}^{-1}$ is low, prediction error $\boldsymbol{\xi}_l = \boldsymbol{\Sigma_l}^{-\frac{1}{2}}(\boldsymbol{\phi}_l - f_l(\boldsymbol{\phi}_{l-1}; \boldsymbol{\theta}_l))$ is low, and will not influence the update of neuron activities and weights.

4 METHODS

Building upon Fontaine & Alexandre (2025), we propose a predictive coding model of the visual cortex and show that it learns hierarchical predictive representations of MNIST images, that support both memory and imagination in a biologically inspired way.

4.1 MODEL

Our model is a PCN with $L = 2$ layers

$$\begin{aligned} p(\boldsymbol{h_0}) &= \mathcal{N}(\boldsymbol{h_0}; \boldsymbol{0}, \boldsymbol{I}) \\ p_{\boldsymbol{\theta}_1}(\boldsymbol{h_1} \mid \boldsymbol{h_0}) &= \mathcal{N}(\boldsymbol{h_1}; f(\boldsymbol{W_1 h_0} + \boldsymbol{b_1}), \boldsymbol{I}) \\ p_{\boldsymbol{\theta}_2}(\boldsymbol{h_2} \mid \boldsymbol{h_1}) &= \mathcal{N}(\boldsymbol{h_2}; \boldsymbol{W_2 h_1}, \boldsymbol{I}) \end{aligned}$$

where $\boldsymbol{\theta}_1 = (\boldsymbol{W_1}, \boldsymbol{b_1})$ and $\boldsymbol{\theta}_2 = \boldsymbol{W_2}$. Indeed, the bias and non-linearity are not required in the input layer, because an image can be represented as a linear combination of basis functions (Olshausen & Field, 1996).

The variational free energy for an input $\boldsymbol{x}$ is

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = \frac{1}{2}\boldsymbol{\xi}_0^\top \boldsymbol{\xi}_0 + \frac{1}{2}\boldsymbol{\xi}_1^\top \boldsymbol{\xi}_1 + \frac{1}{2}\boldsymbol{\xi}_2^\top \boldsymbol{\xi}_2 + \text{constant}$$

where

$$\boldsymbol{\xi}_0 = \boldsymbol{\phi}_0 \tag{4}$$

$$\boldsymbol{\xi}_1 = \boldsymbol{\phi}_1 - f(\boldsymbol{W_1}\boldsymbol{\phi}_0 + \boldsymbol{b_1}) \tag{5}$$

$$\boldsymbol{\xi}_2 = \boldsymbol{x} - \boldsymbol{W_2}\boldsymbol{\phi}_1. \tag{6}$$

4.2 TRAINING ALGORITHM

Let us consider a dataset $\boldsymbol{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ of N i.i.d. samples of a continous variable $\boldsymbol{x}$. The log likelihood can be written $\log p_{\boldsymbol{\theta}}(\boldsymbol{x^{(1)}}, ..., \boldsymbol{x^{(N)}}) = \sum_{i=1}^{N} \log p_{\theta}(\boldsymbol{x}^{(i)})$. Therefore, the variational free energy of the dataset $\boldsymbol{X}$ is

$$\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{X}) = \sum_{i=1}^{N} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x^{(i)}}),$$

which can be estimated based on minibatches

$$\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{X}) \approx \mathcal{L}^{M}(\boldsymbol{\theta}; \boldsymbol{X}^{M}) = \frac{N}{M} \sum_{i=1}^{M} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}^{(i)}; \boldsymbol{x}^{(i)}) \tag{7}$$

where the minibatch $\boldsymbol{X}^{M} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{M}$ is randomly drawn from the dataset $\boldsymbol{X}$.

Given a minibatch $\boldsymbol{X}^{M}$, the latent states $\boldsymbol{\phi}_0^{(i)}$ and $\boldsymbol{\phi}_1^{(i)}$ are updated during T iterations for each datapoint $\boldsymbol{x^{(i)}}$ using equation 2. During training, we do not need to update the latent states until convergence of the variational free energy because we train for multiple epochs, so T can be low to accelerate training. The update rules for the two layers can be calculated from the gradients

$$\begin{aligned}
\nabla_{\boldsymbol{\phi}_1} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) &= -\boldsymbol{W_2}^{\top} \boldsymbol{\xi}_2 + \boldsymbol{\xi}_1 \\
\nabla_{\boldsymbol{\phi}_0} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) &= -\boldsymbol{W_1}^{\top} \operatorname{diag}\left[f'(\boldsymbol{W_1}\boldsymbol{\phi}_0 + \boldsymbol{b_1})\right] \boldsymbol{\xi}_1 + \boldsymbol{\xi}_0.
\end{aligned}$$

Then, the parameters $\boldsymbol{\theta}_1 = (\boldsymbol{W_1}, \boldsymbol{b_1})$ and $\boldsymbol{\theta}_2 = \boldsymbol{W_2}$ are updated once to minimize the estimate $\mathcal{L}^{M}(\boldsymbol{\theta}; \boldsymbol{X}^{M})$ given in equation 7. Thus, the learning rules can be calculated from the sum of gradients $\sum_{i=1}^{M} \nabla_{\boldsymbol{\theta}_l} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{x}^{(i)})$. For an input $\boldsymbol{x}$, it can be shown that

$$\begin{aligned}
\nabla_{\boldsymbol{W_2}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) &= -\boldsymbol{\xi_2} \boldsymbol{\phi_1}^{\top} \\
\nabla_{\boldsymbol{W_1}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) &= -\left[\boldsymbol{\xi_1} \odot f'(\boldsymbol{W_1}\boldsymbol{\phi_0} + \boldsymbol{b_1})\right] \boldsymbol{\phi_0}^{\top} \\
\nabla_{\boldsymbol{b_1}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) &= -\boldsymbol{\xi_1} \odot f'(\boldsymbol{W_1}\boldsymbol{\phi_0} + \boldsymbol{b_1})
\end{aligned}$$

where $\odot$ is the element-wise product. Therefore, we update the parameters with the gradients

$$\begin{aligned}
\nabla_{\boldsymbol{W_2}} \mathcal{L}^{M}(\boldsymbol{\theta}; \boldsymbol{X}^{M}) &\propto \sum_{i=1}^{M} \nabla_{\boldsymbol{W_2}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}^{(i)}; \boldsymbol{x}^{(i)}) = -\sum_{i=1}^{M} \boldsymbol{\xi_2}^{(i)} \boldsymbol{\phi_1}^{(i)\top} \\
\nabla_{\boldsymbol{W_1}} \mathcal{L}^{M}(\boldsymbol{\theta}; \boldsymbol{X}^{M}) &\propto \sum_{i=1}^{M} \nabla_{\boldsymbol{W_1}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}^{(i)}; \boldsymbol{x}^{(i)}) = -\sum_{i=1}^{M} \left[\boldsymbol{\xi_1}^{(i)} \odot f'(\boldsymbol{W_1}\boldsymbol{\phi_0}^{(i)} + \boldsymbol{b_1})\right] \boldsymbol{\phi_0}^{(i)\top} \\
\nabla_{\boldsymbol{b_1}} \mathcal{L}^{M}(\boldsymbol{\theta}; \boldsymbol{X}^{M}) &\propto \sum_{i=1}^{M} \nabla_{\boldsymbol{b_1}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}^{(i)}; \boldsymbol{x}^{(i)}) = -\sum_{i=1}^{M} \boldsymbol{\xi_1}^{(i)} \odot f'(\boldsymbol{W_1}\boldsymbol{\phi_0}^{(i)} + \boldsymbol{b_1})
\end{aligned}$$

During the updates, the gradients are scaled by a multiplicative factor which is the inference rate for the latent states and the learning rate for the parameters. We train the model on a dataset $\boldsymbol{X}$ for multiple epochs, and reduce the learning rate by a multiplicative factor $\gamma < 1$ at each epoch of training to prevent instability issues. Details can be found in section A.1 of the appendix.

4.3 MEMORY REPLAY

Figure 2 shows how this algorithm can be implemented in a neural network mapped onto the visual pathway of the brain, with only local computations. The three levels in the network correspond, from bottom to top, to the lateral geniculate nucleus (LGN) in thalamus (and not the retina because it doesn't receive feedback connections from LGN), the visual cortex (VC) and the entorhinal cortex (EC). This mapping allows us to study memory replay after training the model with algorithm 1.

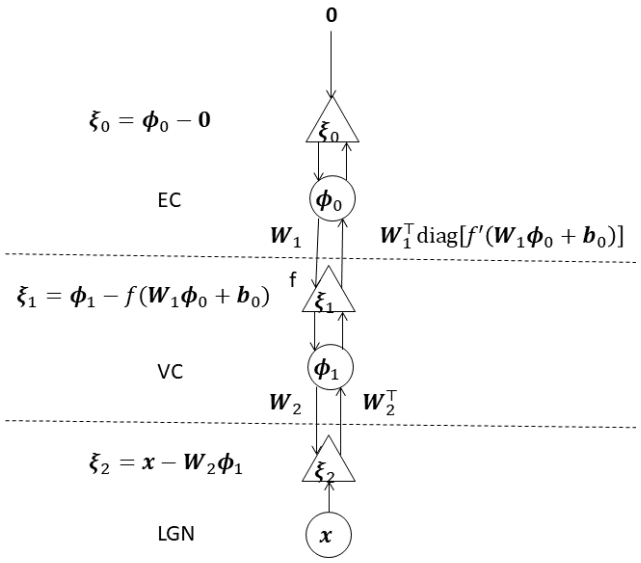


Figure 2: PCN with 3 levels mapped to lateral geniculate nucleus (LGN), visual cortex (VC) and entorhinal cortex (EC). Each node corresponds to multiple neurons. Circle nodes correspond to units ϕ_l and triangle nodes correspond to error units ξ_l. Each connection between nodes corresponds to a fully connected network, with excitatory connections in the upward direction and inhibitory connections in the downward direction. We consider the upmost prediction to be 0, to have a standard normal distribution as prior.

At inference time, the latent states are updated until convergence or until a maximum number of iterations $T_{\max}$ is reached in order to study the converged representations. Convergence occurs when the relative change in the norm of the latent state ϕ_l is smaller than a threshold ϵ, i.e.

$$\frac{\|\nabla_{\phi_l} \mathcal{L}(\theta, \phi; x)\|}{\|\phi_l\|} < \epsilon.$$

During perception, the network is driven by the input image in LGN. While the LGN is set to the image, the VC and EC converge to hierarchical predictive representations of the image. The representation in VC is predictive of the image, as the prediction $W_2\phi_1$ is a reconstruction of the image. The representations in EC can be stored by the hippocampus (not explicitly modelled in this paper) and later replayed during experience replay.

During memory replay, the network is driven by the representation in EC, which is either a representation of an image stored by the hippocampus in the case of experience replay or a sample generated in the latent space of EC in the case of generative replay. While the EC is set to the stored representation, the VC converges to the replayed representation. This representation is protected from ascending input in LGN by setting the precision Σ_2^{-1} in the LGN to 0, preventing the prediction errors in LGN to influence the activity in VC. In this way, attention is focused on the replay, and not on the current input. Then, the prediction $W_2\phi_1$ based on the replayed representation ϕ_1 in VC corresponds to the replayed image.

Sampling of the latent space of EC is class-conditioned. Indeed, we fit a multivariate Gaussian distribution to each class in the latent space of EC, by estimating the mean and covariance of training samples in each class. Then, samples from a given class can be generated by sampling the corresponding Gaussian distribution.

4.4 Experiments

The model is trained on the MNIST dataset, which contains images of handwritten digits from 0 to 9. The original training set of 60,000 images is split into a training set of size 50,000 and a

validation set of size 10,000. The validation set is used to evaluate the model during training and hyperparameter tuning. After training, we evaluate the model on the original test set of 10,000 images.

Hyperparameter values are chosen based on empirical trials informed by the predictive coding literature and summarized in Table 1 in the appendix. In addition, the number of hidden units in level 1 is obtained by minimizing the variational free energy on the validation set for a PCN with $L = 1$ using grid search, as shown in Figure 6 in the appendix. The choice of the number of hidden units in level 2 is more complex and results from a trade-off between different metrics, described in the next subsection.

In our simulations, we study the influence of the number of hidden units in level 0 on the learned representations and on memory replay, both quantitatively and qualitatively. Quantitatively, we evaluate the predictive performance of the model and the quality of the experience replays using the reconstruction and replay errors, based on the root mean squared error (RMSE). The RMSE between two flattened images $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ is

$$\text{RMSE}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \sqrt{\frac{1}{N_{\text{pixels}}} \sum_{i=1}^{N_{\text{pixels}}} (x_i - \hat{x_i}).}$$

The reconstruction error is computed between the original and reconstructed images, averaged over the validation set, whereas the replay error is computed between the original and replayed images, averaged over the training set. In addition, we evaluate the linear separability of the latent manifolds corresponding to the different classes in level 0 of the model using the classification accuracy of a simple multinomial logistic regression. On the qualitative side, we examine examples of reconstructions of images from the validation set and of experience replay corresponding to images from the training set. Additionnally, we show examples of images generated by generative replay and visualize their hierarchical representations.

5 Results

The reconstruction error, replay error and classification accuracy are plotted against the number of hidden units in level 0 in Figure 3. These measures are evaluated after training the model until convergence of all layers, as shown in Figure 7 in the appendix. This figure shows that the second hidden level (counted from the bottom level) in our model does not improve the predictive power of its representations. On the contrary, the reconstruction error increases with the number of units in the second hidden level. However, the quality of replay and the linear separability of the classes in the second hidden level increases with the number of units in that level. Therefore, choosing the number of units in level 0 based on these metrics is not straightforward, and we will turn to the qualitative evaluation.

In Figure 4, we visualize examples of experience replay obtained with models of different widths. As shown in Fontaine & Alexandre (2025), setting the width of the top level to the number of classes (i.e. $n_0 = 10$) results in replayed images that are blurry and that do not retain the details of the original images. This issue is solved by increasing the width of the top level to 30. Increasing it further to 100 improves the sharpness of the replayed images.

However, visual inspection of the reconstructions of images from the validation set and generative replays does not differentiate the models with different widths. Indeed, the difference in reconstruction errors between the different models is imperceptible in the reconstructed images. Similarly, the latent spaces and the quality of the images generated by replay are similar in the different models, despite the difference in replay error and classification accuracy. Therefore, we only show the corresponding plots for a model with $n_0 = 30$ units in level 0 in Figure 5. It can be seen from Figure 5a that the representations learned by the model are perfectly predictive of images it has never seen during training. In addition, according to Figure 5c for the validation set and Figure 8 in appendix for the training set, the representations of the different classes are well separated in all three levels, including the top level which was found to have overlapping clusters in Fontaine & Alexandre (2025). Furthermore, the representations generated by replay mostly fall within the right clusters in all levels. This is confirmed by looking at the generated images in Figure 5b. Most of the images generated for each class are realistic examples of their classes, even though some of them are blurry.

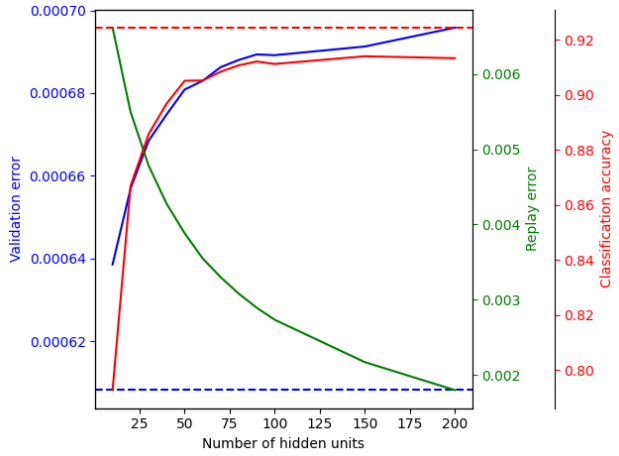


Figure 3: Quantitative evaluation of our model according to different metrics, depending on the number of hidden units in level 0. The dotted lines correspond to the same model but with $L = 1$.

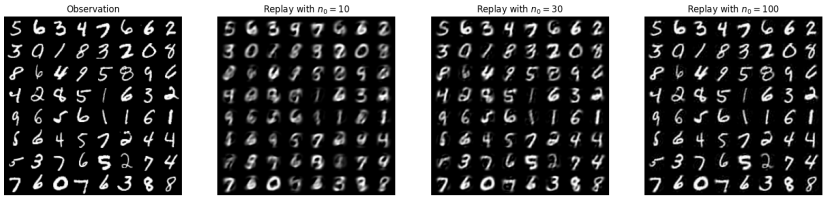
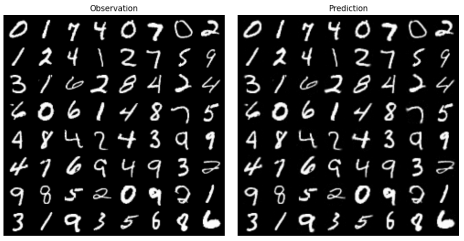


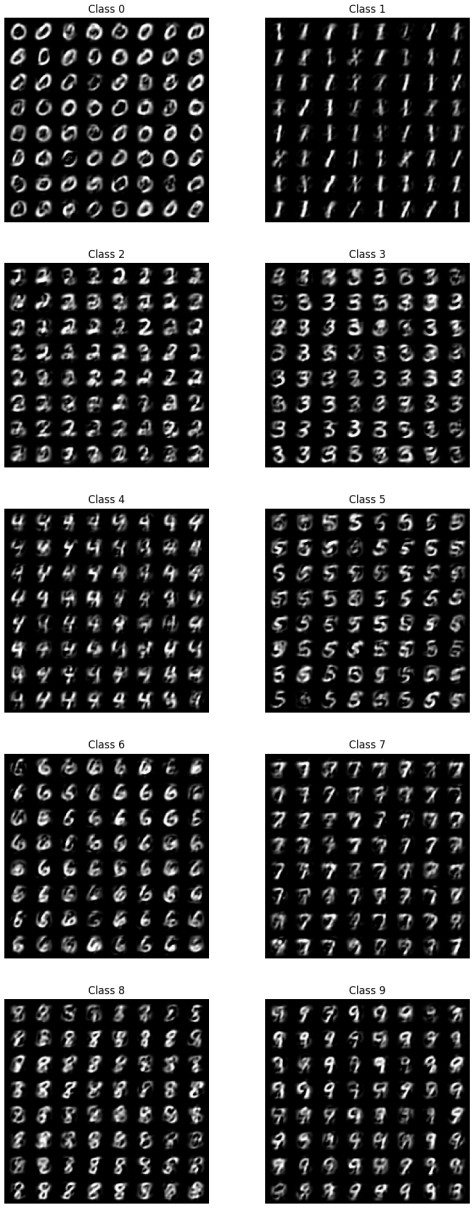
Figure 4: The same minibatch of images from the training set is replayed in PCNs with different widths n_0 at the top level.

6 Discussion

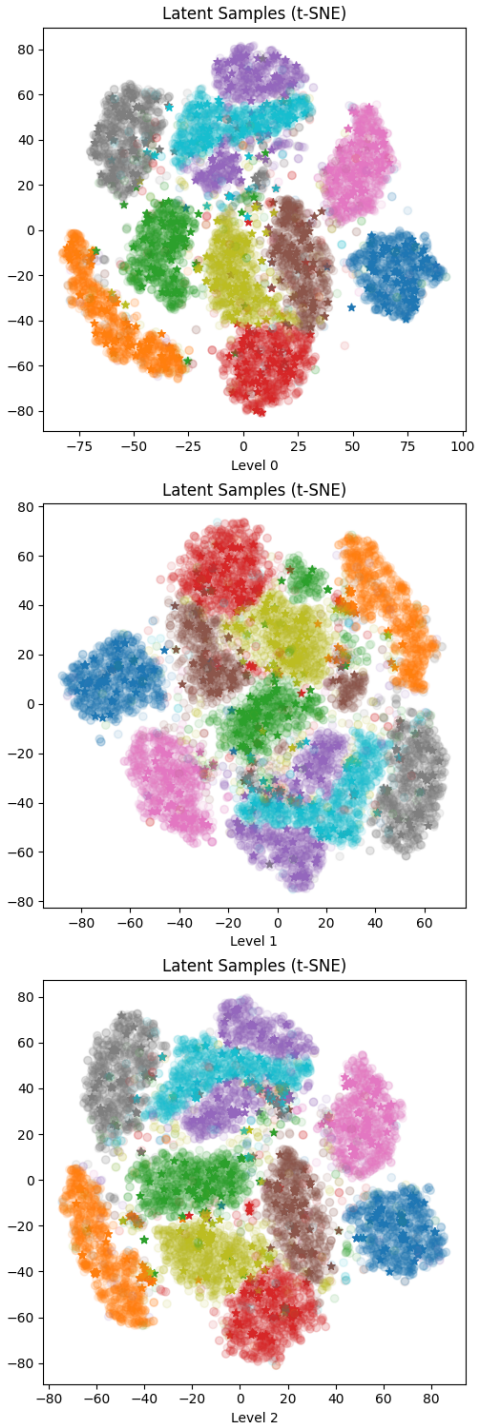
Predictive coding provides a comprehensive account of memory replay, both in the framework of experience replay and generative replay. Generative replay should in practice encompass experience replay as a generative process which samples both existing episodic memories and imagined ones. In this paper, we only provide a proof of concept that predictive coding can account for these two types of replay as we did not completely model the hippocampal formation, but only the entorhinal cortex. Some of the images generated by our model were found to be out-of-distribution, probably because the simple Gaussian distribution we used does not capture the complex, non-linear geometry of the latent manifolds. This issue could be solved using a Riemannian metric (Arvanitidis et al., 2021). However, to provide a more complete, mechanistic account of memory replay, future work should aim at modelling the hippocampal formation with its different components and generating replay from it. In this way, realistic in-distribution samples will naturally be generated thanks to the learned connection between the entorhinal cortex and the hippocampus. Finally, one element that has been left unresolved in this work and that will be important when the hippocampal formation will be connected to the neocortex is the size of the entorhinal layer in the model. Given that the entorhinal cortex is thought to contain a compressed representation combining different sensory modalities and that replay is phenomenologically not an exact copy of episodic memories but is often blurry, a limited width of $n_0 = 30$ could be sufficient.



(a) Reconstructions



(b) Generative replay



(c) Latent and input space

Figure 5: Visualizations for a model with $n_0 = 30$ units in the top level. (a) Images reconstructed (right) by a PCN with $n_0 = 30$ units in the top level for a random mini-batch of images from the validation set (left). (b) Images generated by replay for each class. (c) Hierarchical representations of the images generated by replay (star-shaped markers) and of the images of the validation set (transparent circle markers), visualized in 2D using t-SNE. Each image is represented as one data point in each of the three subplots, colored according to the class.

REFERENCES

Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models, 2021. URL https://arxiv.org/abs/1710.11379.

Helen C Barron, Ryszard Auksztulewicz, and Karl Friston. Prediction and memory: A predictive coding account. *Progress in neurobiology*, 192:101821, 2020.

György Buzsáki. Hippocampal sharp wave-ripple: A cognitive biomarker for episodic memory and planning. *Hippocampus*, 25(10):1073–1188, 2015. doi: https://doi.org/10.1002/hipo.22488. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/hipo.22488.

Harriet Feldman and Karl Friston. Attention, uncertainty, and free-energy. *Frontiers in Human Neuroscience*, 4, 2010. ISSN 1662-5161. doi: 10.3389/fnhum.2010.00215. URL https://www.frontiersin.org/journals/human-neuroscience/articles/10.3389/fnhum.2010.00215.

Lucie Fontaine and Frédéric Alexandre. Semantic and episodic memories in a predictive coding model of the neocortex. In *2025 International Joint Conference on Neural Networks*, Rome, Italy, June 2025. URL https://hal.science/hal-05202850.

Karl Friston. Learning and inference in the brain. *Neural Networks*, 16(9):1325–1352, 2003.

Antonino Greco, Marco D'Alessandro, Karl J. Friston, Giovanni Pezzulo, and Markus Siegel. Sensory robustness through top-down feedback and neural stochasticity in recurrent vision models, 2025. URL https://arxiv.org/abs/2508.07115.

Jeff Hawkins, Marcus Lewis, Mirko Klukas, Scott Purdy, and Subutai Ahmad. A framework for intelligence and cortical function based on grid cells in the neocortex. *Frontiers in Neural Circuits*, Volume 12 - 2018, 2019. ISSN 1662-5110. doi: 10.3389/fncir.2018.00121. URL https://www.frontiersin.org/journals/neural-circuits/articles/10.3389/fncir.2018.00121.

Dharshan Kumaran, Demis Hassabis, and James L McClelland. What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in cognitive sciences*, 20 (7):512–534, 2016.

Beren Millidge. Implementing predictive processing and active inference: Preliminary steps and results. March 2019. doi: 10.31234/osf.io/4hb58. URL http://dx.doi.org/10.31234/osf.io/4hb58.

Gaspard Oliviers, Rafal Bogacz, and Alexander Meulemans. Learning probability distributions of sensory inputs with monte carlo predictive coding. *PLOS Computational Biology*, 20(10): 1–34, 10 2024. doi: 10.1371/journal.pcbi.1012532. URL https://doi.org/10.1371/journal.pcbi.1012532.

Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

Alexander Ororbia and Daniel Kifer. The neural coding framework for learning generative models. *Nature Communications*, 13:2064, 04 2022. doi: 10.1038/s41467-022-29632-7.

Luca Pinchetti, Chang Qi, Oleh Lokshyn, Cornelius Emde, Amine M'Charrak, Mufeng Tang, Simon Frieder, Bayar Menzat, Gaspard Oliviers, Rafal Bogacz, Thomas Lukasiewicz, and Tommaso Salvatori. Benchmarking predictive coding networks – made simple. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=sahQq2sH5x.

Eleanor Spens and Neil Burgess. A generative model of memory construction and consolidation. *Nature Human Behaviour*, pp. 1–18, 2024.

Ivilin Stoianov, Domenico Maisto, and Giovanni Pezzulo. The hippocampal formation as a hierarchical generative model supporting generative replay and continual learning. *Progress in Neurobiology*, 217:102329, 2022. ISSN 0301-0082. doi: https://doi.org/10.1016/j.pneurobio.2022.102329. URL https://www.sciencedirect.com/science/article/pii/S0301008222001150.

Gido M. van de Ven, Hava T. Siegelmann, and Andreas Savas Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature Communications*, 11, 2020. URL https://api.semanticscholar.org/CorpusID:221111120.

Renhao Wang, Kevin Frans, Pieter Abbeel, Sergey Levine, and Alexei A. Efros. Prioritized generative replay, 2025. URL https://arxiv.org/abs/2410.18082.

A Appendix

A.1 Algorithm

The algorithm described in section 4.2 is summarized in algorithm 1 where α, β are the inference and learning rates. The constant $\frac{N}{M}$ from equation 7 is factorized in the learning rate β. To prevent instabilities which occured systematically during training, we propose an exponential learning rate scheduler

$$\beta_{\text{epoch}} = \gamma \times \beta_{\text{epoch}-1}$$

which decays the learning rate β by a multiplicative factor γ at each epoch.

Initialization parameters include the standard deviations $\sigma_{\boldsymbol{W}}$ and $\sigma_{\boldsymbol{\phi}}$ and the number of dimensions of latent state $\boldsymbol{\phi_0}$.

Algorithm 1 Model training with minibatches

$\boldsymbol{W_1}, \boldsymbol{W_2} \leftarrow$ Sample from $\mathcal{N}(0, \sigma_{\boldsymbol{W}})$
$\boldsymbol{b_1} \leftarrow \mathcal{U}(-\frac{1}{n_0}, \frac{1}{n_0})$
repeat
  for $k = 1$ to n_{batches} **do**
    $\boldsymbol{X}^M \leftarrow$ Random minibatch of M datapoints drawn from $\boldsymbol{X}$
    for $i = 1$ to M **do**
      $\boldsymbol{\phi_2^{(i)}} \leftarrow \boldsymbol{x}^{(i)}$
      $\boldsymbol{\phi_0^{(i)}}, \boldsymbol{\phi_1^{(i)}} \leftarrow$ Sample from $\mathcal{N}(0, \sigma_{\boldsymbol{\phi}})$
      $\boldsymbol{\xi_0^{(i)}}, \boldsymbol{\xi_1^{(i)}}, \boldsymbol{\xi_2^{(i)}} \leftarrow$ Calculate the corresponding errors (equation 4)
      for $t = 1$ to T **do**
        $\boldsymbol{\phi_1^{(i)}} \leftarrow \boldsymbol{\phi_1^{(i)}} + \alpha(\boldsymbol{W_2}^\top \boldsymbol{\xi_2^{(i)}} - \boldsymbol{\xi_1^{(i)}})$
        $\boldsymbol{\phi_0^{(i)}} \leftarrow \boldsymbol{\phi_0^{(i)}} + \alpha(\boldsymbol{W_1}^\top \text{diag}\left[f'(\boldsymbol{W_1}\boldsymbol{\phi_0^{(i)}} + \boldsymbol{b_1})\right] \boldsymbol{\xi_1^{(i)}} - \boldsymbol{\xi_0^{(i)}})$
        $\boldsymbol{\xi_0^{(i)}}, \boldsymbol{\xi_1^{(i)}}, \boldsymbol{\xi_2^{(i)}} \leftarrow$ Calculate the corresponding errors (equation 4)
      end for
    end for
    $\boldsymbol{W_2} \leftarrow \boldsymbol{W_2} + \beta \sum_{i=1}^{M} \boldsymbol{\xi_2^{(i)}} \boldsymbol{\phi_1^{(i)\top}}$
    $\boldsymbol{W_1} \leftarrow \boldsymbol{W_1} + \beta \sum_{i=1}^{M} \left[\boldsymbol{\xi_1^{(i)}} \odot f'(\boldsymbol{W_1}\boldsymbol{\phi_0^{(i)}} + \boldsymbol{b_1})\right] \boldsymbol{\phi_0^{(i)\top}}$
    $\boldsymbol{b_1} \leftarrow \boldsymbol{b_1} + \beta \sum_{i=1}^{M} \boldsymbol{\xi_1^{(i)}} \odot f'(\boldsymbol{W_1}\boldsymbol{\phi_0^{(i)}} + \boldsymbol{b_1})$
  end for
until variational free energy $\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{X})$ is minimized

A.2 Experiments

Table 1: Hyperparameter values

Parameter	Value
Activation function f	$\tanh$
Batch size	64
Standard deviation $\sigma_{\boldsymbol{W}}$	0.01
Standard deviation $\sigma_{\boldsymbol{\phi}}$	0.05
Number of iterations T_{train}	50
Number of iterations T_{valid}	200
Maximum number of iterations T_{max}	20000
Convergence threshold ϵ	2×10^{-4}
Inference rate α	0.01
Inference optimizer	SGD
Initial learning rate β_0	10^{-5}
Learning rate decay factor γ	0.99
Learning optimizer	Adam

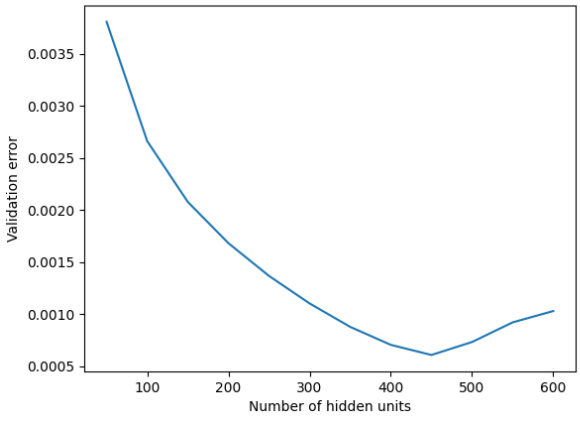


Figure 6: RMSE between original and reconstructed images averaged over the validation set for a PCN with $L = 1$.

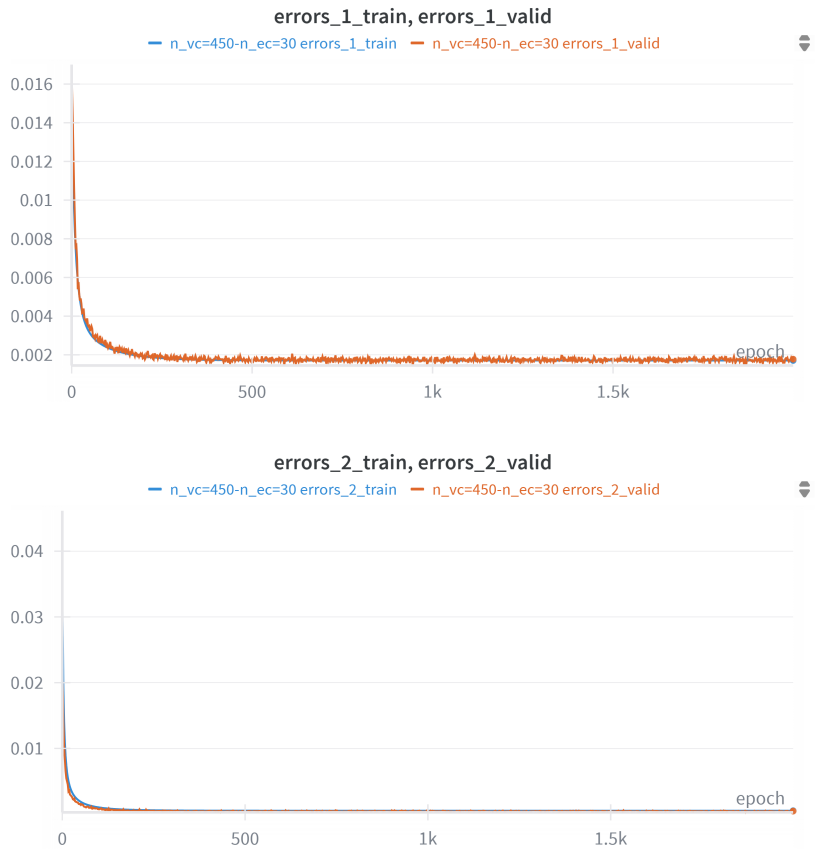


Figure 7: Mean prediction errors in level 1 and 2 of a model with 30 units in level 0, averaged over the whole training set (blue line) and a random minibatch from the validation set (orange line) at each epoch. The model converges after 2000 epochs thanks to the learning rate scheduler.

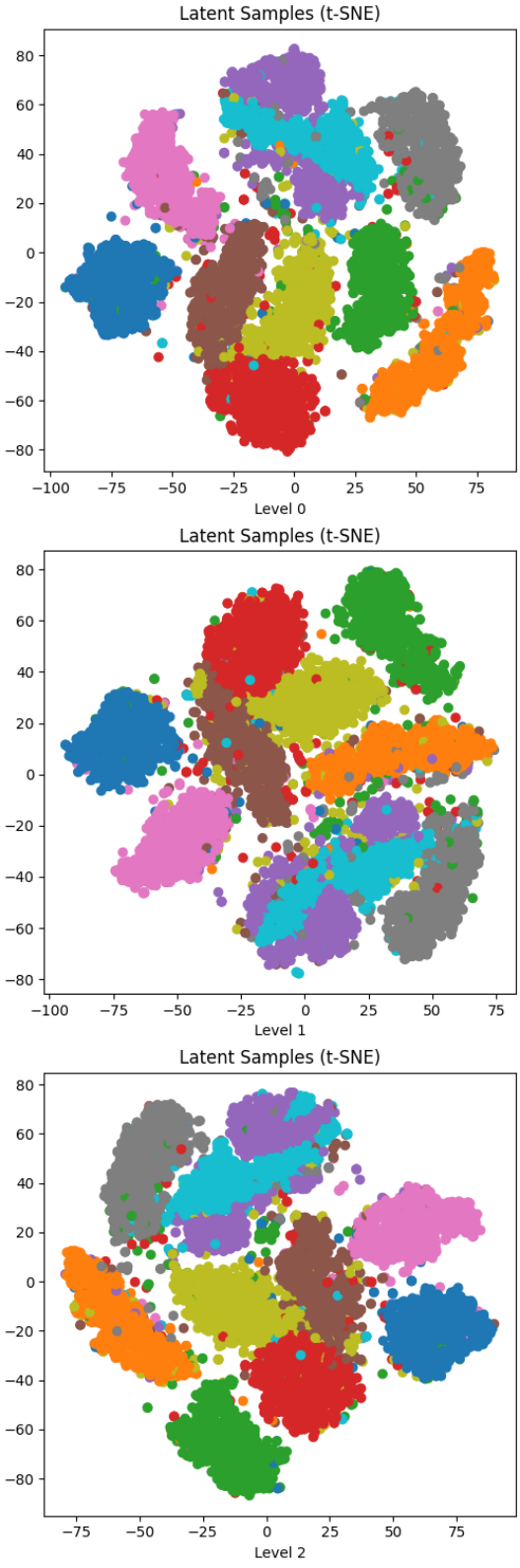


Figure 8: Hierarchical representations of the training images visualized using t-SNE. Each image from the training set is represented as one data point in each of the three subplots, colored according to the class.